

NAVAL POSTGRADUATE SCHOOL MONTEREY, CALIFORNIA



THESIS

DESIGN OF ADVANCED ANALYSIS SOFTWARE FOR IT- 21 COMPLIANT NETWORKS

by

David R. DeMille

December 1999

Thesis Advisor:
Co-Advisor:

John McEachen
Murali Tummala

Approved for public release; distribution is unlimited.

20000313 039

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188.	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE December 1999	3. REPORT TYPE AND DATES COVERED Master's Thesis		
4. TITLE AND SUBTITLE DESIGN OF ADVANCED ANALYSIS SOFTWARE FOR IT-21 COMPLIANT NETWORKS		5. FUNDING NUMBERS		
6. AUTHOR(S) David R. DeMille				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSORING / MONITORING AGENCY REPORT NUMBER		
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) The cornerstone to achieving battlefield dominance, as defined by Joint Vision 2010, is establishing and maintaining information superiority. This new paradigm of network centric warfare has shifted computer networking from an administrative support system to a tactical necessity. Recognizing this critical requirement for interoperable networking, the Department of the Navy promulgated the Information Technology for the Twenty-First Century (IT-21) computing standards. This thesis investigates the synergetic interactions between application software, the Windows™ NT operating system, and the underlying networks. The insight gained is then exploited to develop performance analysis software in C++. The resulting application provides a valuable asset for examining, troubleshooting, and optimizing IT-21 information systems.				
14. SUBJECT TERMS Asynchronous Transfer Mode, ATM, IT-21, WINSOCK			15. NUMBER OF PAGES 63	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

**DESIGN OF ADVANCED ANALYSIS SOFTWARE FOR IT-
21 COMPLIANT NETWORKS**

David R. DeMille
Lieutenant, United States Navy
B.S.E.T., Texas A&M University, 1991

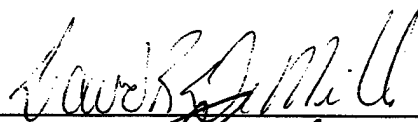
Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING-

from the

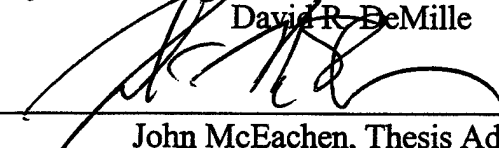
**NAVAL POSTGRADUATE SCHOOL
December 1999**

Author:

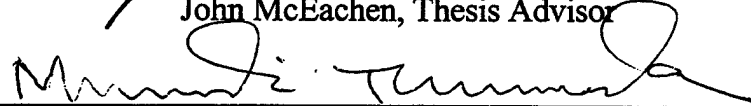


David R. DeMille

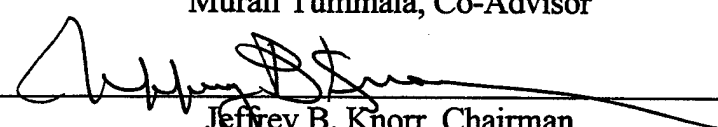
Approved by:



John McEachen, Thesis Advisor



Murali Tummala, Co-Advisor



Jeffrey B. Knorr, Chairman

Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The cornerstone to achieving battlefield dominance, as defined by Joint Vision 2010, is establishing and maintaining information superiority. This new paradigm of network centric warfare has shifted computer networking from an administrative support system to a tactical necessity. Recognizing this critical requirement for interoperable networking, the Department of the Navy promulgated the Information Technology for the Twenty-First Century (IT-21) computing standards. This thesis investigates the synergetic interactions between application software, the Windows™ NT operating system, and the underlying networks. The insight gained is then exploited to develop performance analysis software in C++. The resulting application provides a valuable asset for examining, troubleshooting, and optimizing IT-21 information systems.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION.....	1
A. THESIS OBJECTIVE.....	1
B. THESIS ORGANIZATION.....	2
II. BACKGROUND	3
A. IT-21 NETWORK TOPOLOGY	3
B. WINDOWS™ NT ARCHITECTURE	5
1. Basic Structure	5
2. The Registry	6
3. Communication Sub-Structure.....	7
4. Network Sockets	8
C. IT-21 SYSTEM ANALYSIS REQUIREMENTS	8
III. THE DESIGN.....	11
A. DEVELOPMENT ENVIRONMENT	11
B. BASIC STRUCTURE	13
C. CONFIGURATION INFORMATION.....	14
1. Using The Registry Networking Entries.....	15
2. Implementation Overview	16
3. Data Collection and Handling.....	16
4. GUI Design	16
D. NETWORK ACCESS AND TIMING	18
1. Timing and Timers.....	19
2. Socket Support	20
E. COMMUNICATIONS CONTINUITY AND CONNECTIVITY	20
1. ICMP and IP	21
2. Implementation Issues	21
3. GUI	21
F. BENCHMARKING.....	23

IV. RESULTS AND PERFORMANCE	27
V. CONCLUSION	33
A. SUMMARY OF WORK.....	33
B. FUTURE WORK.....	33
APPENDIX A. INFORMATION TECHNOLOGY FOR THE TWENTY-FIRST CENTURY (IT-21).....	35
APPENDIX B. HUNGARIAN NOTATION	39
APPENDIX C. CLASS OVERVIEW	41
LIST OF REFERENCES	45
INITIAL DISTRIBUTION LIST	47

LIST OF FIGURES

Figure 1: IT-21 In Naval Networking	4
Figure 2: NT Structure the operating system.	6
Figure 3: The OSI Model and Windows™ NT	7
Figure 4: General Application Structure	14
Figure 5: Network Adapter and Driver Registry Entries	15
Figure 6: Installed Protocols GUI Layout	17
Figure 7: Installed Adapters GUI Layout.....	18
Figure 8: Connectivity and Continuity GUI Layout.....	22
Figure 9: Benchmarking GUI Layout	24

THIS PAGE INTENTIONALLY LEFT BLANK

ACRONYMS AND ABBREVIATIONS

API	Application Programming Interface
ATM	Asynchronous Transfer Mode
COTS	Commercial Off-the-Shelf
DON	Department of the Navy
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HAL	Hardware Abstraction Layer
IP	Internet Protocol
IS	Information Systems
IT-21	Technology for the 21st Century
LAN	Local Area Network
LLC	Logical Link Control
MAC	Media Access Control
MFC	Microsoft Foundation Classes
NDIS	Network Device Interface Specification
NIC	Network Interface Card
NSAP	Network Service Access Point
OSI	Open Systems Interconnection
PVC	Permanent Virtual Circuit
QoS	Quality of Service
SNMP	Simple Network Management Protocol
SPI	Systems Programming Interface
SVC	Switched Virtual Circuit
TCP	Transport Control Protocol
UDP	User Datagram Protocol
WSA	Windows Sockets API
WSC	Windows Socket Configuration

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENT

I would like to take this opportunity to recognize the outstanding support I received with this thesis. The glory and credit belong to the Lord Jesus Christ for giving me the opportunity to attend NPS and bringing friendly, helpful people into my life. To both professors John McEachen and Murali Tummala, I really appreciate the time and support you sacrificed for me from your busy schedules. To all the Fore Systems technical support representatives, thank you for the prompt and helpful assisting in implementing the Fore ATM service provider interface. And to the administrative support staff, with special emphasis to Alice Lee and Sue Netzorg, your outstanding support was instrumental in making my tour of duty at Naval Postgraduate School enjoyable and successful.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

On March 30, 1997, the U.S. Navy promulgated a new policy for all Department of the Navy (DON) information systems (IS): Information Technology for the 21st Century (IT-21). [1] IT-21 mandates Windows™ NT based hosts on fast Ethernet local area networks (LANs) connected by asynchronous transfer mode (ATM) backbones.

With the increasing emphasis on network centric warfare and battlefield information dominance, the performance and reliability of the Navy's IT-21 systems is critical to the ability to effectively engage in operations. This reliance on IS gives the network support personnel a direct, critical role in the success of warfare missions.

In order to effectively support operations, these administrators need to configure, monitor, optimize, and test networks quickly and effectively. There are many commercial off-the-shelf (COTS) tools available to assist network support personnel, hereafter referred to as administrators. However, most of these tools are very expensive and many operate on only a small sub-set of the installed network components. This thesis will focus on designing a tool to provide a wide range of services to administrators across all the IT-21 networking standards.

A. THESIS OBJECTIVE

This thesis will design, develop, test and analyze the performance of an IT-21 compliant network analysis software package. The primary objective is to design fully functional network analysis software that can be easily upgraded and expanded. Developing this software serves three primary purposes:

1. Provides a valuable tool to the fleet.
2. Provides a useful utility to the advanced networking lab at the Naval Postgraduate School (NPS) in support of other networking research.
3. Documents how protocols and networking services are implemented in Windows™ NT with emphasis on how to properly use them in network application development and performance analysis.

B. THESIS ORGANIZATION

This thesis is organized as follows: Chapter II covers the foundational information underlying both IT-21 application and analysis principles. It begins by discussing IT-21 network topology as implemented in the fleet and then briefly covers the Windows™ NT implementation details significant to network performance. It concludes by deriving the required functionality for a comprehensive IT-21 configuration and analysis utility.

Chapter III explicitly details how each functional component is designed. It will emphasize both the implementation method and the internal details of Windows™ NT that apply.

Chapter IV will provide analysis of performance. Chapter V will summarize work accomplished and provide ideas for follow-on work.

Appendix A is a copy of the IT-21 Naval message. Appendix B will detail the Microsoft™ recommended Hungarian naming convention used in this project, and is extremely useful when reading the Microsoft™ examples and sample code. Appendix C contains a description of each class developed for this project.

II. BACKGROUND

The client computers and underlying networks that comprise the DON IS assets provide an invaluable cohesive communications service. Optimizing the end-to-end performance of this service, not that of each single networking component is a predominate DON objective. Therefore, optimization techniques must balance a wide set of requirements. In addition to high-speed connectivity, the critical nature of military operations requires a very high degree of reliability. Further complicating system optimization, the mobile, maritime network participants introduce a uniquely dynamic communication topology. Finally, when optimizing current performance, the administrator needs to retain the flexibility to innovate, conduct research, and field upgrades in order to maintain a maximum state of readiness.

While the primary goal of both network engineers and administrators is to optimize communications reliability and speed to the user, the process varies for each network topology. Each physical implementation and protocol provides its own unique suite of requirements. In addition, each client operating system must also be included in the optimization process because its implementation of transport protocols directly relates to users' observed network performance. In short, network engineers and administrators would benefit greatly from a single utility designed to analyze each network component and provide a complete, coherent representation of the network's current condition. Therefore, the IT-21 network topology is a fundamental basis to the design of a compressive network analysis utility to support the Navy's networking needs.

A. IT-21 NETWORK TOPOLOGY

To increase interpretability and reduce support requirements, the IT-21 specification sets standards for all hardware, software, and LAN technology employed in DON systems. The three primary requirements that effect data communications performance are: Windows™ NT 4.0, Fast Ethernet LANs, and ATM backbones. This straightforward specification works well at established, shore stations; however, the naval networks required

in support of real-world operations are much more complicated because of their inherent dependence on dynamic composition, participant mobility, secure encryption, and wireless communication links. A simplified configuration for a typical naval network is shown in Figure 1. In addition to the aforementioned challenges, naval communications networks integrate voice traffic, support legacy networks, and must integrate with non-compliant networks like the naval tactical data system (NTDS).

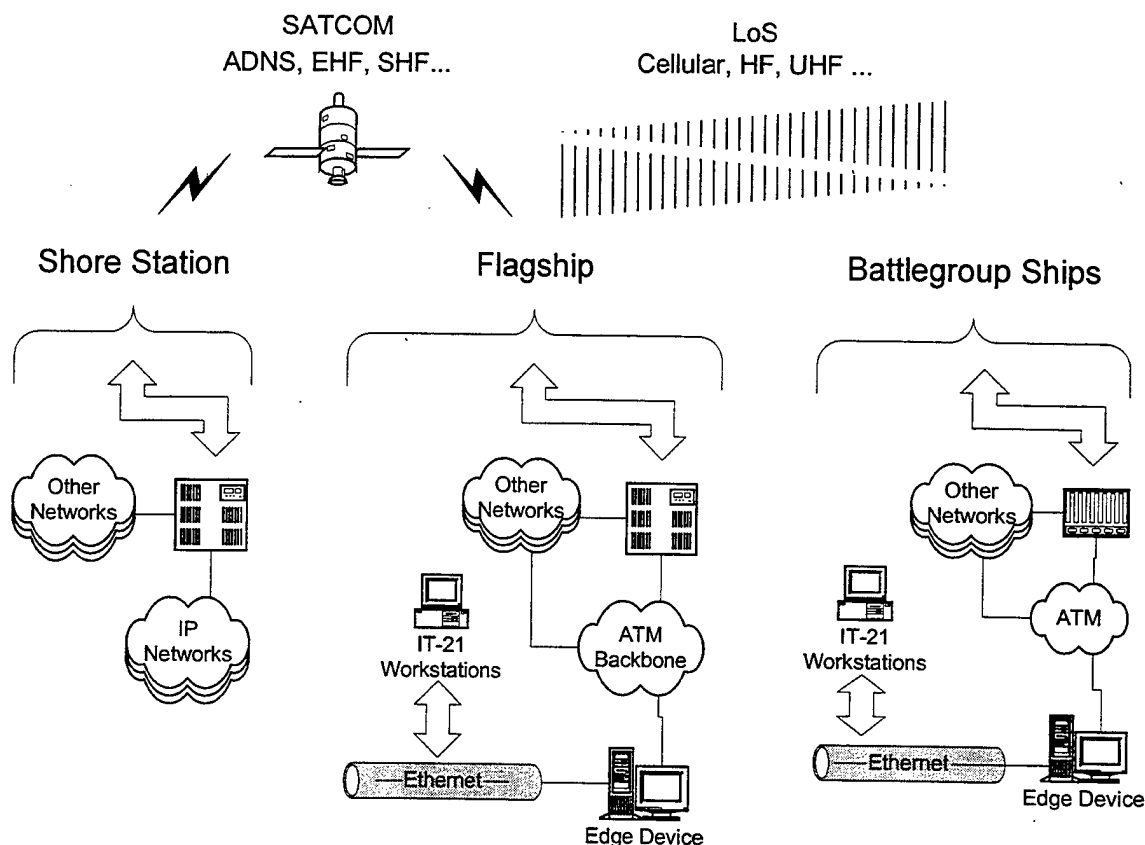


Figure 1: IT-21 In Naval Networking

In order to ensure reliable networking services, mobile units need the ability to monitor and measure individual performance of each organic networking component (client computers, Ethernet LAN segments, ATM pipes) and their combined performance. In addition, the ability to obtain performance information about the inorganic networking links to remote sites increases the ability to pinpoint communications bottlenecks and breakdowns, increasing reliability by facilitating coordinating between network administrators.

The IT-21 architecture requires an edge device between the Ethernet and ATM network segments. A Windows™ NT server with multiple network interface cards (NICs) can serve as a cost effective edge device, but may introduce troubleshooting difficulties.

B. WINDOWS™ NT ARCHITECTURE

A fundamental basis to IT-21 system performance is the networking implementation inherent in the required Windows™ NT operating system. Windows™ NT is designed as a secure, modular computing system, which profoundly impacts its structure, implementation, use, and interpretation. Its design also has serious implications for the networking support implementation. Other significant design aspects include service providers, sockets, timers, and computing precision.

1. Basic Structure

Windows™ NT is foundationally designed according to the concept of a reference monitor. There are three criteria a reference monitor must satisfy:

- Absolute control—the reference monitor must completely control all access to hardware and software components.
- Isolation—the kernel must be a separate, self-contained entity protected from alteration or outside influences.
- Verifiability—the kernel must demonstrate it enforces the required security policies.

The NT kernel executes in *privileged* mode and controls all access to networking and timing assets. Software applications execute in *protected* mode and request access to assets from the kernel using standardized application programming interface (API) calls.

In keeping with the principle of isolation, the kernel is also separated from the installed hardware by the hardware abstraction layer (HAL). Device drivers are the software

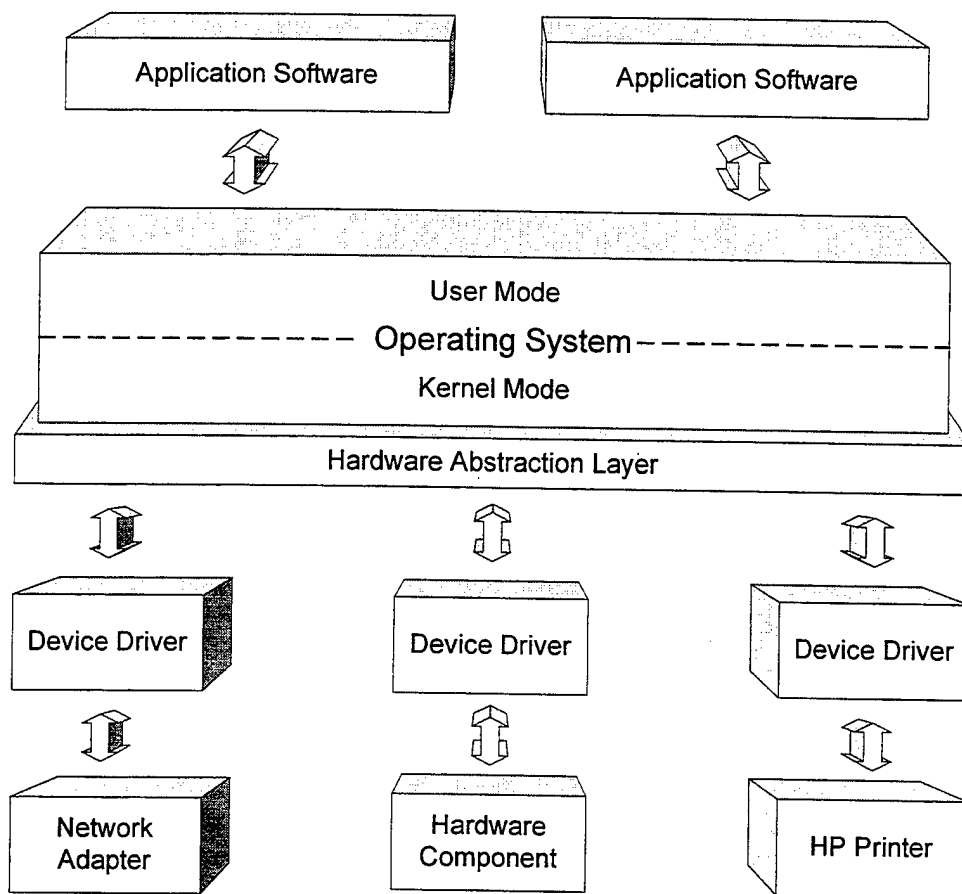


Figure 2: NT Structure the operating system

components that directly control hardware[◊] and must provide the HAL required interface to This standard interface allows the NT kernel cross-platform portability. An overview the NT architecture is shown in Figure 2.

2. The Registry

One important byproduct of the HAL and of the multi-level security scheme is Windows™ NT relies on a database of configuration information to interface with its hardware. During boot-up NT scans the computer's buses and hardware ports for installed

[◊] This does not hold for printing assets. Microsoft™ refers to printing hardware as print devices. The drivers for print devices are called printers. (e.g. The HP 4000 printer is the software driver to control an HP 4000 print device.)

hardware and checks to insure that there is an entry for each in the registry. Duplicate, obsolete, and superfluous registry entries not detected.

The registry is structured much like the computer's file system, but instead of directories and files it has *keys* and *values*. Keys can contain other keys and values. Values can consist of character strings, DWORDs (32 bits), or binary data.

3. Communication Sub-Structure

The open systems interconnect (OSI) communications model is often used to conceptualize how communications should function. Figure 3 shows how the OSI model relates to the NT networking implementation. [2] In addition to the OSI model, Figure 3 also shows the relationship to the user and kernel operating modes.

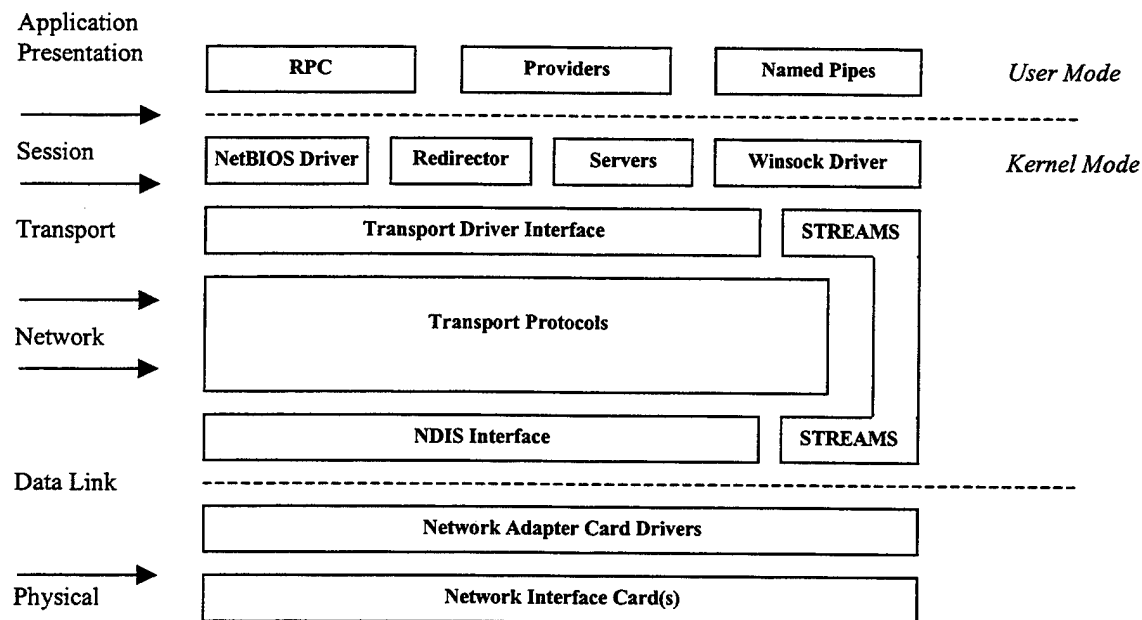


Figure 3: The OSI Model and Windows™ NT

At the application level, user applications execute in protected mode. These applications interface with the presentation level through API calls to the NT kernel. The kernel supports the session layer through calls to dynamic linked libraries (DLLs). The transport and network layers and their interfaces correspond to the service providers in NT.

While protocols are platform independent specifications, their implementation is unique and affects their compatibility. Windows™ NT implements protocols and supports name resolution with *service providers*. It is important to understand the distinctions of the terminology: *MSAFD Tcpip* is the Microsoft™ supplied service provider that implements the functionality for the TCP, UDP, and IP protocols. Therefore, protocol names are used when referring to properties native to the protocol specification; service provider names are used when referring to implementation issues.

The data link layer is composed of the logical link control (LLC) and media access control (MAC) sub-layers. The LLC functions above the HAL and is implemented with the network device interface specification (NDIS) 3.0, a joint Microsoft™ and 3Com™ specification. The MAC is implemented with device drivers; therefore, it operates below the HAL.

4. Network Sockets

The most important communications programming mechanisms provided by Windows™ NT are Windows Sockets (Winsock), which are based on Berkeley software distribution (BSD) sockets. [2] A socket is a communications endpoint and is either blocking or non-blocking. Blocking sockets force the calling application to wait while they execute; non-blocking sockets return control to the calling program immediately.

Winsock is implemented in both 16- and 32-bit modes. In its first incarnation, Winsock 1.0 (winsock.dll), only the TCP/IP suite of protocols was supported. Winsock 2.0 (ws2_32.dll) includes support for other protocols (ATM) and implements quality of service (QoS) support. The Winsock functions execute in privileged mode and provide only the access allowed to the calling program's security level.

C. IT-21 SYSTEM ANALYSIS REQUIREMENTS

The IT-21 IS suite is a complete system comprised of many functional components, including client computers, local networks, and communications links. In this system, users run client software, which interfaces with the NT kernel to access network assets and communicate. The entire system must work in harmony to maximize rapid, reliable access

to information. While application responsiveness and proper GUI design can significantly enhance the user's perceived performance, the network engineer is typically more concerned with performance, precision, and accuracy. Because the performance afforded to the user must be optimized, the ability to test all networking components from a single application and correlate those results into a coherent picture of system performance would be an invaluable asset. To convey a complete understanding of system performance, a network analysis tool should provide detailed configuration descriptions, useful testing and verification utilities, and measure performance benchmarks.

Because accessing networking assets relies on the configuration information stored in the registry, the registry networking entries are crucial in troubleshooting, optimizing, and developing distributed applications. While Windows™ NT includes two utilities, *regedit* and *regedt32*, to display and edit the registry, both are too unwieldy and unintuitive to provide a practical method for viewing network configurations. Therefore, networking analysis software should begin by consolidating and displaying the networking entries in a legible format. Configuration information for all networking assets should be included: network adapters, networking drivers, and installed protocols. The bindings between adapters and protocols can be viewed and edited from *Control Panel* → *Network*, making their inclusion unnecessary.

Networking analysis software should also be able to verify network end-to-end connectivity, trace network routes, and isolate trouble spots. Windows™ NT includes the two ubiquitous command line utilities, *ping* and *tracert*, which support these functions for TCP/IP networks. This software will include enhanced versions of these utilities.

In order to analyze and optimize IT-21 networking services, Ethernet, ATM, and composite performances must be measured. Providing benchmark measurements of the end-to-end connections at the transport layer will provide a good indication of composite performance. Complete networking analysis also requires the ability to measure the performance of each network segment at the network layer. This thesis will implement this ability for ATM adaptation layer (AAL) 5, best effort traffic because it is the most common class of traffic on the network backbones.

Network analysis software should be modular to support maintainability, upgrading and expansion; incorporate multi-threading to improve usability and allow benchmarking in execution contexts; and follow the Microsoft™ recommended GUI guidelines. The measurement uncertainties due to preemptive multi-tasking must be accounted for to provide the most accurate results possible.

III. THE DESIGN

While this networking analysis software is a valuable tool for both network administration and research, examining the design process and code structure provides valuable insights into networking performance theory and implementation issues. Software applications are built on the foundations provided by the software development tools used to create them. Implementing networking support efficiently is essential to all network applications and is especially critical in performance analysis. Performance measurements are also dependant on the accuracy and precision of the timers used. Developing robust utilities to determine network status and configurations provides a good foundation for the more advanced and difficult benchmarking routines.

A. DEVELOPMENT ENVIRONMENT

There are many programming languages and development environments available to build software applications. Despite the recent hype over Java™, the increased flexibility, robustness, and innate power of C++ make it the language of choice for developing advanced networking applications. This thesis uses the Microsoft™ Visual C++ suite for code development.

Windows™ NT applications can be developed to run in protected mode using a software development kit (SDK) or in privileged mode using a device driver development (DDK) kit. Privileged execution provides direct access to the installed network hardware components, allowing advanced functionality like ATM cell capture and Ethernet promiscuous mode. Unfortunately, this unrestricted access incurs several high costs, including lack of code portability, increased programming complexity, and debugging difficulties. Because this mode of execution operates below the Winsock DLL, socket support is not available for these advanced functions.

Protected mode execution relies on the NT kernel for access to network services. This restricts functionality to those services supplied by the NT operating system or the installed hardware drivers. While protected mode programming precludes providing cell

capture and other abilities requiring direct, low-level access, it is capable of providing precise and accurate performance estimates. Because the NT operating system provides sufficient functionality to measure networking performance and implement many network analysis utilities, programming in the more difficult privileged mode is not required. In addition, protected mode programming produces portable, easily maintained code.

When using Visual C++, application code for Windows™ NT interfaces with the operating system through either the raw Windows™ application programming interface (API) or the Microsoft™ foundation classes (MFC). The MFC libraries encapsulate much of the Windows™ API, providing an easier programming interface at the expense of additional overhead. The MFC libraries are designed to support embedding raw Windows™ API function calls; therefore, there is no good reason not to use the MFC API.

Microsoft™ strongly recommends using the document/view architecture and has implemented MFC's support for many of an application's basic features in the document/view functions. [6] The document/view architecture exploits the object oriented programming (OOP) features of C++ and conceptually separates an application into document objects and view objects. Document objects, derived from *CDocument*, read, write, and control access to data. View objects, derived from *CView*, display data and allow the user to select or edit data. While *CDocument* functionality is not necessary, the robust display tools in the MFC view classes make the document/view architecture an ideal development framework.

MFC document/view based applications use either single document interface (SDI) or multiple document interface (MDI) windows. The capability to handle more than one document at a time is not required for this design. However, this application uses MDI because of its ability to display multiple child windows simultaneously.

The MFC libraries, Windows API, and sample source code incorporate Hungarian notation, a programming convention named after the nationality of its creator: Microsoft™ programmer Charles Simonyi. In its simplest form, Hungarian notation advocates appending lowercase prefixes to variable names to indicate their type. The fundamental purpose of Hungarian notation was to increase the legibility of C code. Unfortunately, different dialects

have formed over time (at least four partially incompatible dialects just within Microsoft™).
[6] This thesis will use the common dialect outlined in Appendix B.

B. BASIC STRUCTURE

Network analysis software must collect and display information about the available networking assets and must employ those assets to provide networking services. Both of these functions require extracting information from the local computer. Therefore, the design begins with a *CDocument* derived object to collect all the available networking related information from the local computer during initialization. This will allow the application to be easily upgraded to support storing configuration information to disk for reference purposes and accessing remote computer's networking configurations.

Networking support is implemented through an object encapsulating socket support. This object also includes timing support and returns execution times for functions. By including the timing support with the network access object, the code design becomes more modular.

Each major application activity is implemented through a *CView* derived object. These objects encapsulate both the GUI and the required computational functionality. The code structure provides for easy expansion. Figure 4 provides a graphical representation of the application structure.

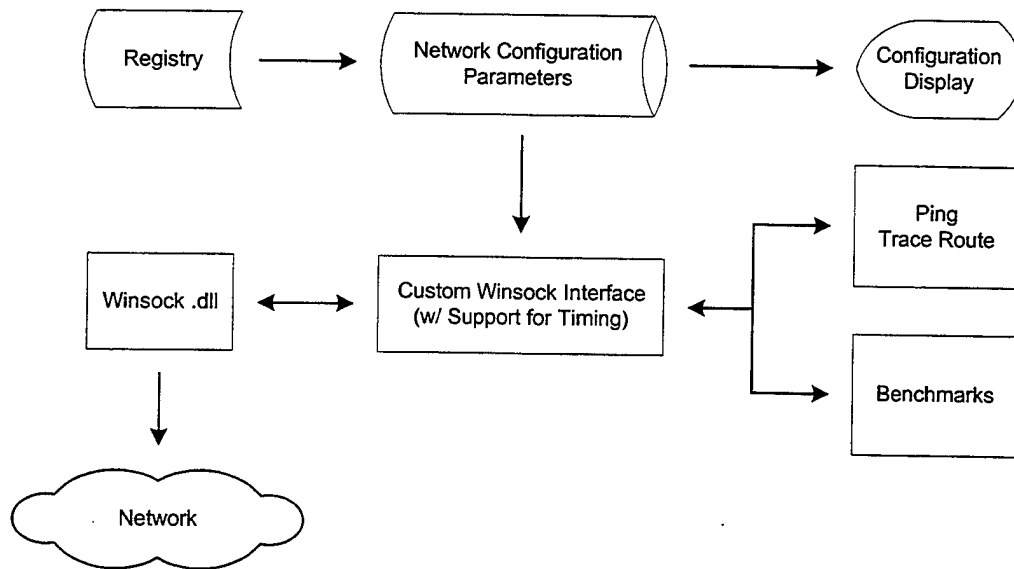


Figure 4: General Application Structure

C. CONFIGURATION INFORMATION

As mentioned earlier, Windows™ NT stores its networking configuration information on adapters, drivers, protocols, and bindings in the registry. Because the accuracy of this information directly affects networking performance, the ability to access it is significant to administrators, network engineers, and network programmers. Improper, duplicate, or obsolete networking registry entries can degrade observed networking performance, invalidating analysis software results.

1. Using The Registry Networking Entries

Accessing the registry requires opening one of the keys, reading or writing values under that key, then closing the key. A limited number of keys can be open at any one time; therefore, keys should be released as quickly as possible. The Windows™ API provides a suite of functions to manipulate registry entries; MFC encapsulates registry manipulation in the *CRegKey* class. Figure 5 illustrates how the network adapter and driver entries are stored in the registry.

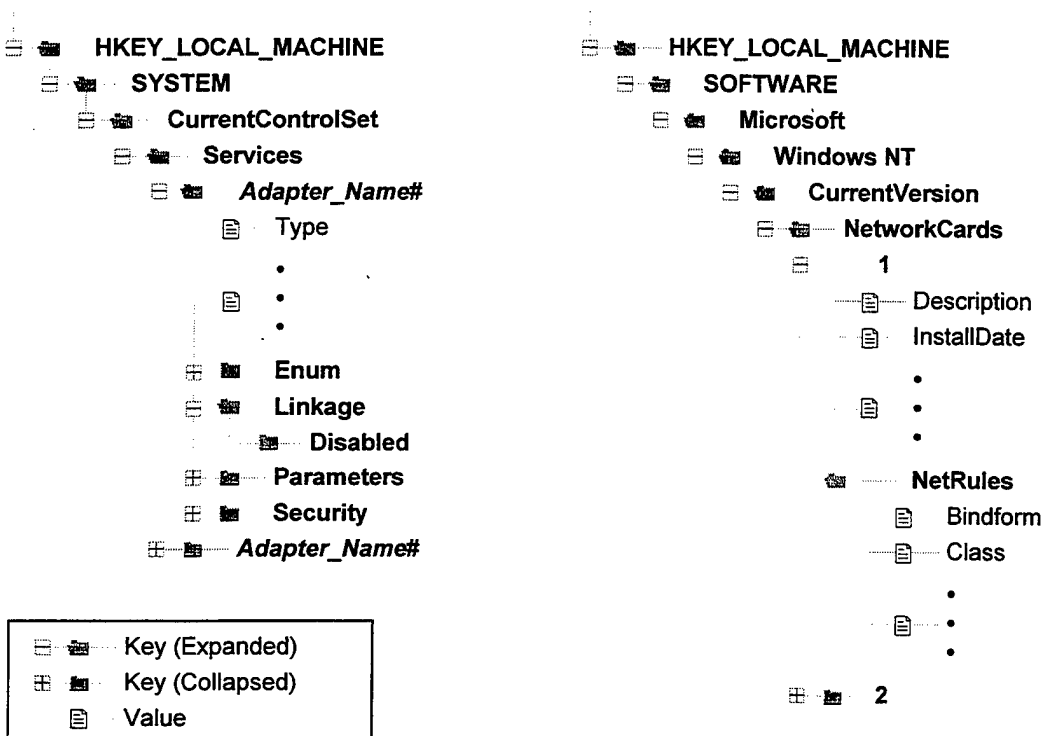


Figure 5: Network Adapter and Driver Registry Entries

The entries for network adapters and their drivers are not grouped in a single location, but are dispersed in the registry. In addition, the configuration for both adapters and drivers are listed under the `...\\Services\\Adapter_Name#\\` key. Therefore, at least two keys are required for each NIC, typically `...\\Services\\Adapter_Name\\` and `...\\Services\\Adapter_Name1\\`. The *Type* value identifies whether the entry is for the adapter (0x04) or its driver (0x01). The `...\\NetworkCards\\#\\` keys are linked to the corresponding `...\\Services\\Adapter_Name#\\` entries by their `...\\#\\ProductName` and `...\\#\\ServiceName` values (not explicitly shown in Figure 5).

While protocol configuration parameters are also spread across several distributed registry keys, the Win32 API provides two functions to retrieve this information: *WSAEnumProtocols(...)* and the more robust *WSCEnumProtocols(...)*.

2. Implementation Overview

During initialization, this network analysis package will instantiate a *CDocument* derived object to obtain the networking configuration of the local machine. The installed adapter and protocol parameters will then be formatted and displayed in a simple, straightforward manner using *CView* derived objects. The GUI should epitomize completeness and clarity.

3. Data Collection and Handling

The object used to collect and manage the network configuration data was derived from *CDocument* to provide *CView* coordination through inheritance. It uses three private arrays to store the parameters: protocol parameters, network adapters, and network drivers. The inherited persistence (disk storage) support is not used, but could be exploited in future versions to save, compare, and verify the networking registry entries.

4. GUI Design

It seems reasonable to display the adapter and protocol configurations in separate windows. In both cases, the analysis software must list the installed components, clearly demonstrate their relationships with each other, and detail the configuration and information associated with each. After much trial-and-error, I found the optimum format for organizing all this information in a single interface was to split a window into halves: The left half composed of a tree control, allowing the user to select from a list of the installed components; the right half displaying the configuration information for the selected component.

MFC supports split windows with *CSplitterWindow*. This class is derived directly from *CWnd*; therefore, it lacks *CView* functionality. During initialization, *CView* derived objects are inserted as static *panes* in the splitter window (this implementation prohibits using dynamic panes). It is straightforward to derive a class from MFC's *CTreeView* to

provide the necessary support for the tree control. However, there is more information to be displayed in the right pane than there is room.

Windows™ provides *tab controls* to manage multiple dialogs. The combination of a tab control and its dialogs is often referred to as a *property sheet*. MFC encapsulates this behavior with *CPropertySheet* and *CPropertyPage*. Property sheets provide a convenient, efficient way to partition and display large amounts of information.

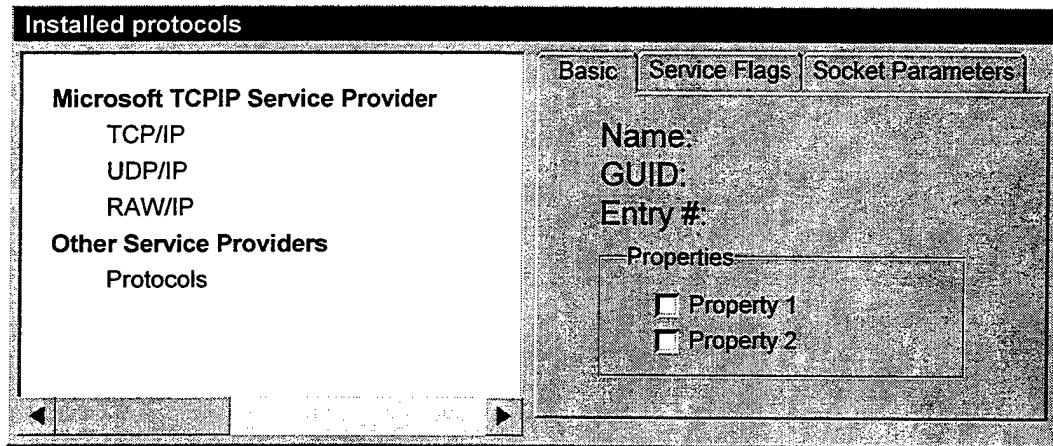


Figure 6: Installed Protocols GUI Layout

Figure 6 shows the layout for the installed protocols GUI. The protocols are grouped in the tree control according to their service provider. The property sheet displays the information corresponding to the selected node in the tree control. This information is grouped into three sections: basic information, service flags, and socket specific parameters. The basic information property page will include the protocol name, globally unique identifier (GUID), catalog entry number^o, and provider flags. The next property page lists the 19 service support flags, including guaranteed delivery, broadcast, and QoS. The third property page contains the parameters needed to instantiate a socket for this protocol.

^o This number indicates the service provider's relative priority and is used to determine default behavior. Priorities are set based on installation order and can be changed with the *sporder.exe* utility included with Winsock 2.

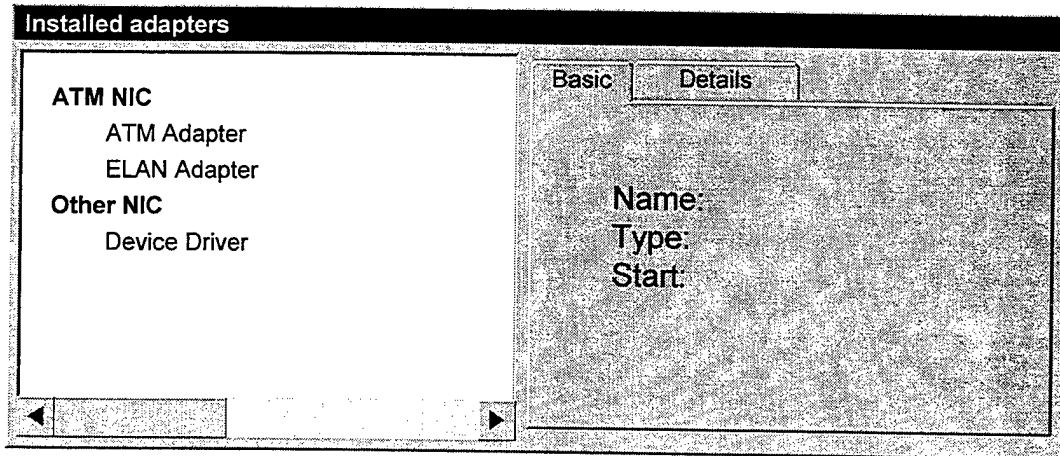


Figure 7: Installed Adapters GUI Layout

The installed adapters GUI layout in Figure 7 is almost identical to the installed protocols GUI layout. However, the dialog associated with the *Details* tab in this GUI will depend on the selection in the tree control. If a hardware component is selected, a dialog for that type of hardware will list its settings. For example, if an Ethernet NIC is selected, its Ethernet address, media type, and other settings will be displayed along with universal parameters of interrupt, slot number, etc. An ATM NIC, on the other hand, will include buffer sizes, cell rates, and other ATM settings.

Bindings control how network adapters and protocols interface with each other. Windows™ NT supports viewing and editing bindings under *Settings→Control Panel→Network→Bindings*. Because a convenient interface with the bindings is provided, there is no compelling reason to display the bindings in the networking analysis package.

D. NETWORK ACCESS AND TIMING

In order to function, network analysis software must establish realistic communications and accurately measure the performance of those connections. Because timing support is included in this network access implementation, timing events in Windows NT will be examined first. Then, adding timing and measurement capabilities to the windows sockets in support of networking application development and performance benchmarking will be explored.

1. Timing and Timers

Because we are interested in measuring performance, accurately timing networking events is critical. The NT kernel offers three primary ways to time events: timers, high-performance timers, and performance counters. Standard timers have a resolution on the order of milliseconds, making them unfit for performance benchmarking. High-performance timers are implemented with performance counters.

Performance counters are 64-bit values stored in the registry that are monotonically incremented, typically by device drivers. These counters can be used to measure many rates of performance. The performance monitor in Windows™ NT allows installed performance counters to be selected and displayed, including processor, networking, and system tasks. While the performance monitor can measure many network events[◊], it reports the aggregate activity of installed components not the performance of the attached network or specific communications channels.

The Windows™ API returns performance counter values as *LARGE_INTEGER* unions. This union allows the values to be manipulated as either 64-bit integers or as two 32-bit integers. The Visual C++ compiler includes native support for 64-bit integers using the *LONGLONG* type.

Windows™ NT provides precision timing through the *QueryPerformanceCounter()* and *QueryPerformanceFrequency()* API functions. The performance frequency on Intel™ based computers typically provides timing resolution on the same order as the processor clock speed.

The precision of high-resolution timing results gives a false sense of accuracy. Each performance counter measurement may include observation noise. Windows™ NT is a preemptive multitasking system and may pause the current thread execution without warning. In addition, access to the registry, memory, and other system resources are queued for servicing, introducing random processing delays.

[◊] The Network Monitor service must be loaded to obtain TCP/IP statistics.

2. Socket Support

Sockets provide the communications support necessary for networking support, but do not include the timing for analysis. Socket communications can be developed using the native Winsock API calls or a MFC socket class. *CAsyncSocket* encapsulates the Winsock functionality. It requires the programmer to account for network byte order differences, multi-byte character sets (MBCSs), and other low-level issues. *CSocket* is derived from *CAsyncSocket* and accounts for these issues, at the cost of additional overhead.

Because efficiency and execution time is critical when trying to accurately measure events, this project's performance-aware socket class is derived from *CAsyncSocket* and overrides its methods so they conform to the following pseudo code procedure:

- Variable initialization and translation
- Start = performance timer
- *CAsyncSocket::ThisFunction()*;
- Stop = performance timer
- Return (Stop – Start)

For example, substituting *WSAConnect()* or *::Connect()* for *CAsyncSocket::ThisFunction()* gives an implementation to measure call setup time. By only measuring the duration of the actual communications function, the effect of the conditions in the local machine are minimized. This provides the most likely estimate of the actual service rate provided by the underlying networks and is an upper limit to network performance a user can expect on that machine.

Microsoft™ products are notorious for containing bugs. I found that the Winsock 2.0 *receive()* family of functions fail to properly timeout when called from a user thread. This is probably because the active window improperly hooks the *WM_TIMER* message generated by the timeout. In any case, this situation can be avoided by using *select()* to wait for incoming data. Once *select()* verifies valid data is queued, the *receive()* functions can safely retrieve it.

E. COMMUNICATIONS CONTINUITY AND CONNECTIVITY

Networking performance cannot be measured without reliable communications. Two command-line utilities, *ping* and *tracert*, exploit internet control message protocol

(ICMP) to verify connectivity and determine the routing path. This networking analysis package will include enhanced versions of these utilities. While this thesis only implements these utilities using with ICMP, it could be expanded to provide the same functionality for other type of networks using the same interface (determining the routing path of a VPI/VCI pair would be a useful improvement).

1. ICMP and IP

ICMP is typically considered part of the IP layer and is explained in detail by Stevens in [3]. At its most basic level, ICMP is used to send control and error messages across an IP based network. An ICMP *echo request* causes the target network node to retransmit the packet's contents back to the transmitting station. This is a simple and convenient method to verify connectivity.

By also using the time-to-live (TTL) field in the IP header, the maximum number of hops can be set. Each time an IP packet is routed, its TTL is decremented. When the TTL reaches zero, the packet is discarded and an ICMP *time exceeded* message is sent back to the transmitter. By successively sending IP packets with increasing TTL values, the transmitting station will receive ICMP messages from each routing point in the path to the target station.

2. Implementation Issues

At the IP layer, implementing *ping* and *tracert* is straightforward, almost trivial. Unfortunately, Winsock does not provide direct access to the IP layer. Sockets of type RAW/IP do allow most of the IP header fields to be set via Winsock API functions. However, only processes with administrator privileges are allowed to instantiate RAW sockets. Therefore, this network analysis software will provide connectivity and continuity support only to users with administrator access on the host machine.

3. GUI

One way to improve over the available *ping* and *tracert* utilities is to fully leverage the advantages of the Windows™ NT graphical environment. The general layout for the connectivity and continuity test dialog is shown in Figure 8. Each of its sections is discussed below.

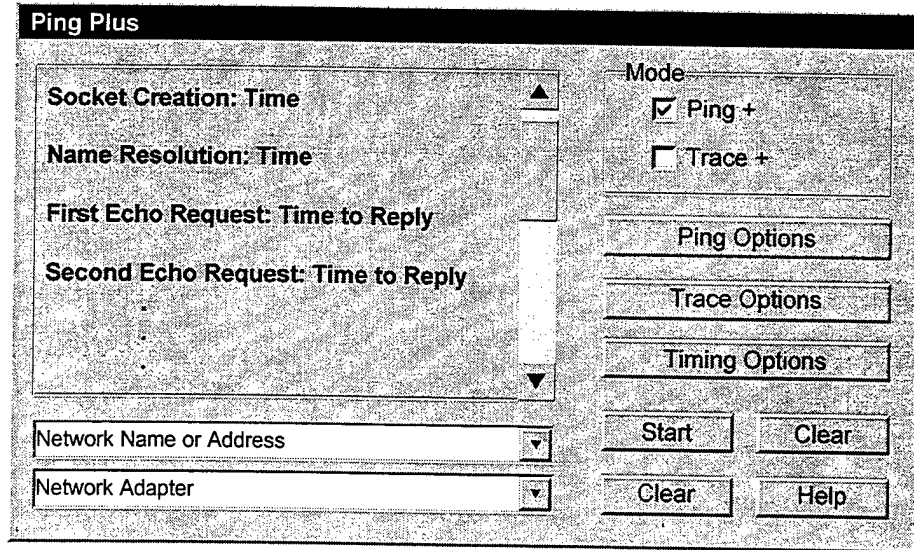


Figure 8: Connectivity and Continuity GUI Layout

One significant advantage over the command-line ping utility is the ability to select the network adapter with which to test the network. The user typically has no control of which NIC is used in a multi-NIC computer. The *drop-down list box* control allows the user to choose from all installed adapters[◊], even though not all adapters support ICMP (e.g. native ATM adapter). This list could be filtered to show only adapters that support ICMP; however, limiting information in this context seems counterintuitive.

A *drop-down combination box* control is used to allow the user to either type in the target address or select one from history. The 20 previous targets are stored in the registry in registry and are loaded during program initialization. Because target addresses are typically used more than once, this feature is both convenient and timesaving.

The *text* control formats and displays results according to the user-selected options. It includes a *slider* so previous results can be viewed after they have scrolled off the screen. This control uses a 64k buffer for text, limiting it to about 800 lines—more than sufficient. The user also has the option of clearing the control of text. The control is set as read only,

[◊] The adapter list provided by NT is really a list of the installed drivers. Typically, Ethernet NICs have only one driver. However, ATM NICs will have at least two, one for ATM and another for IP over ATM.

preventing user alteration; its window *inactive* attribute is set, preventing selection and protecting cursor placement.

The output is customizable, allowing the user to select which event(s) to time and display from the following list:

- Socket Creation
- Name Resolution
- Reply (Always reported)
- Timeout

Measuring the duration of operations that timeout helps to verify the operation of the local timeout mechanism. The user can also select whether the times are displayed in milliseconds or microseconds and the number of significant digits to report.

Because of the close relationship between verifying continuity and determining connectivity, both functions are implemented in this single interface. The user selects the active operation mode with the *radio button* control.

F. BENCHMARKING

The primary measure of network performance is data throughput. This analysis suite provides performance benchmarks by establishing a communications session with a remote host, transmitting data, and measuring the transmission times. The results must be displayed in an intuitive format, accurately reflecting network performance. The user interface should also be generic enough to apply to different types of networks. The general format of the GUI is shown in Figure 9.

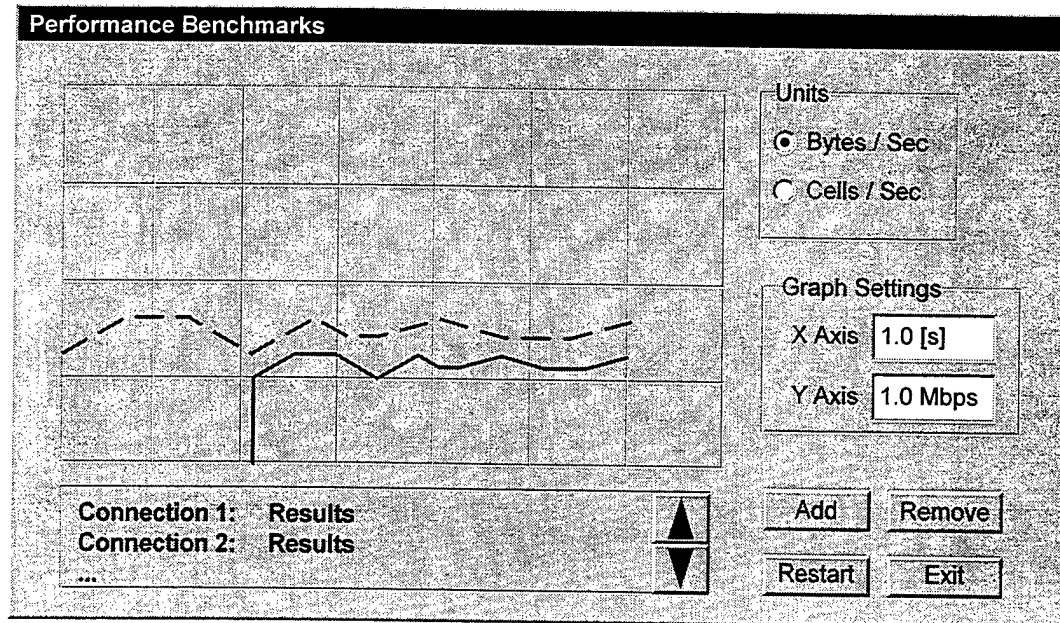


Figure 9: Benchmarking GUI Layout

Each of the current benchmarks being conducted will be listed in a text control at the bottom of the dialog. This control will list the connection id (VPI/VCI for ATM, TCP port number for TCP/IP, etc.), the latest measured throughput, and the average throughput over the life of the connection. In addition, the text color will match the color of the trace in the performance plot.

The graph will plot the performance measurements of each running connection. The time axis runs from left to right and uses a user-defined resolution. The vertical axis will represent the units selected by the user and may represent different measures for different session. For example, an ATM connection may be displayed in cells per second alongside a TCP/IP connection displayed in bytes per second. When the results exceed the maximum index on the time axis, the plot is scrolled to the left by one time resolution. This results in new measurements being added at the far right of the plot so the full time range of values can be seen. Old measurements are discarded as they scroll off the left edge of the graph.

A separate dialog is used to establish benchmark connections. This allows the software to use custom dialogs to obtain the information needed for each type of connection. This permits using the same GUI to display results for different types of connections. This consistent interface simplifies use and facilitates future expansion.

This GUI is very robust; however, not all its capabilities are used. The GUI can support multiple concurrent benchmarks, but the underlying measurement implementation cannot. To measure the performance of multiple connections accurately, each connection must run in its own thread of execution. The groundwork for multi-threaded support is laid, but it is not fully functioning at this time. Also, only TCP/IP and ATM AAL5 connections are supported.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS AND PERFORMANCE

While developing this network analysis software, the underlying fundamental design criteria have remained relatively unchanged. However, the GUI layouts have continually evolved as experience has revealed improved approaches. The following is a description of the first version of this software package: NetMan 1.0. In addition to providing a guide to operation, this will provide a baseline to measure future editions against.

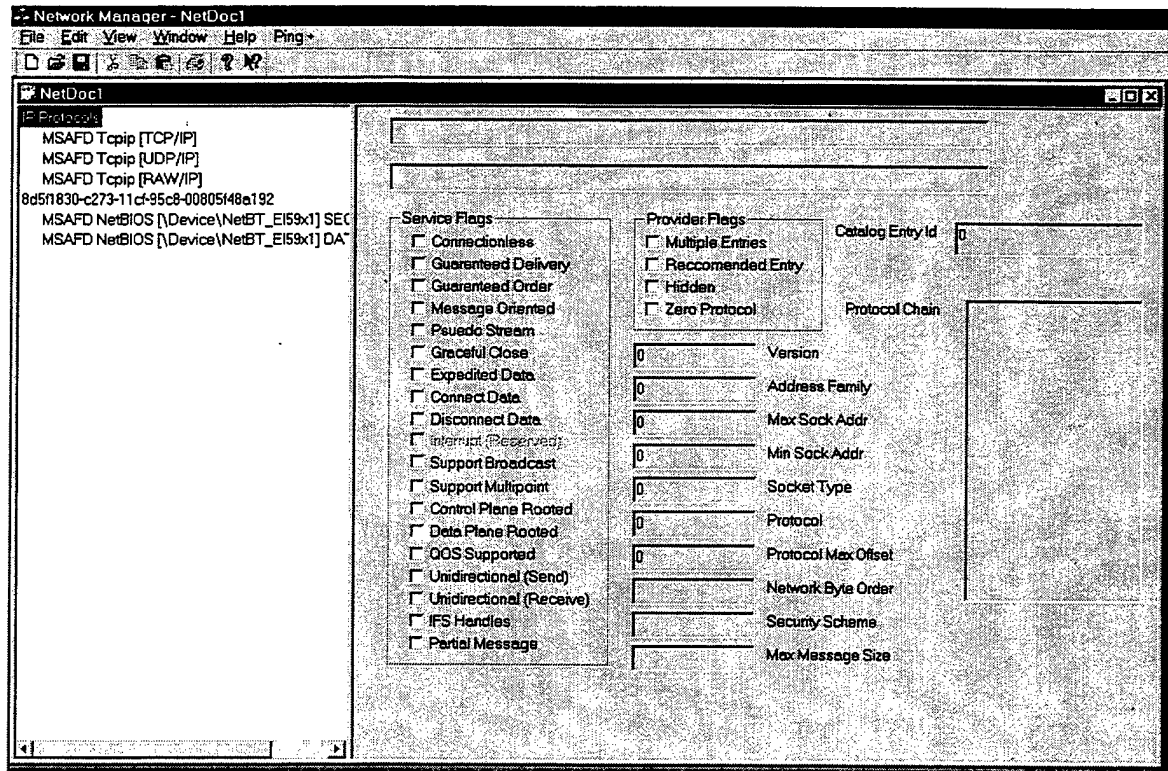


Figure 10: Service Providers/Protocols GUI 1

The first incarnation of the installed protocols GUI is shown in Figure 10. This GUI works on monitors running at resolutions of 1200x1600. Because this GUI only worked well on 21" monitors at this high resolution, the GUI in Figure 11 was developed.

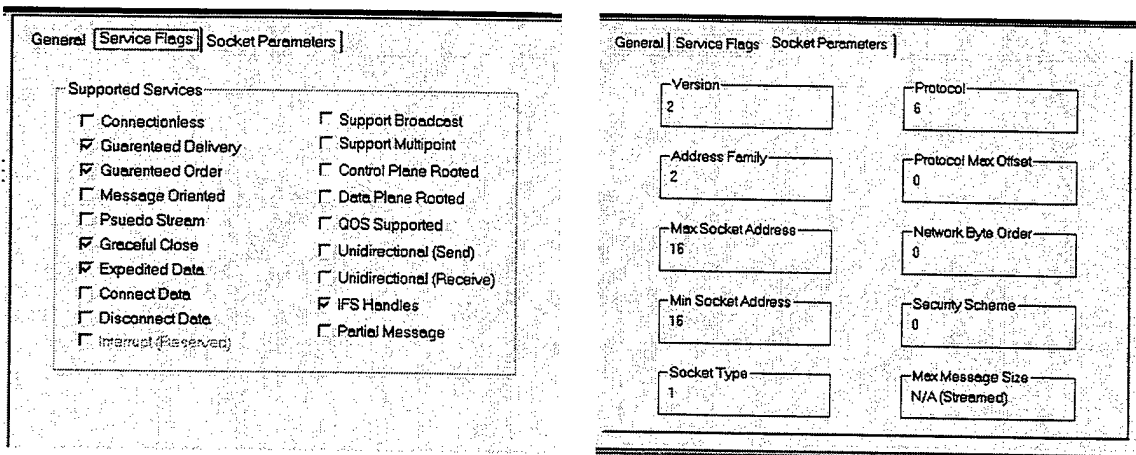
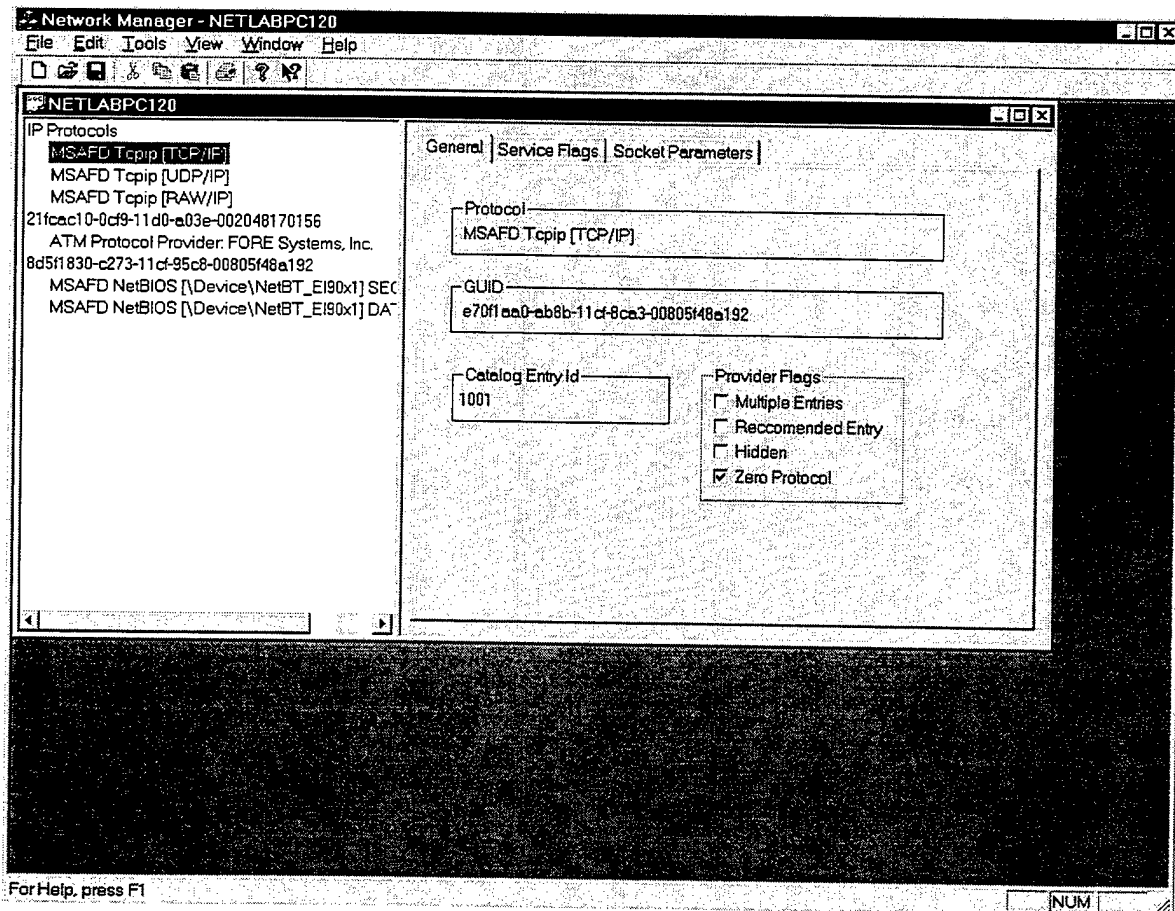


Figure 11: Service Providers/Protocols GUI 2

The connectivity and continuity testing GUI and its options GUIs are shown in Figure 12. In this example, the ping operation used the local Ethernet NIC; the trace operation used the ATM NIC. With all events selected in the Reporting dialog, socket creation/binding to local address and name resolution times are measured and displayed. Because this version only implements connectivity and continuity tests using ICMP, the selected ATM adapter must support TCP/IP.

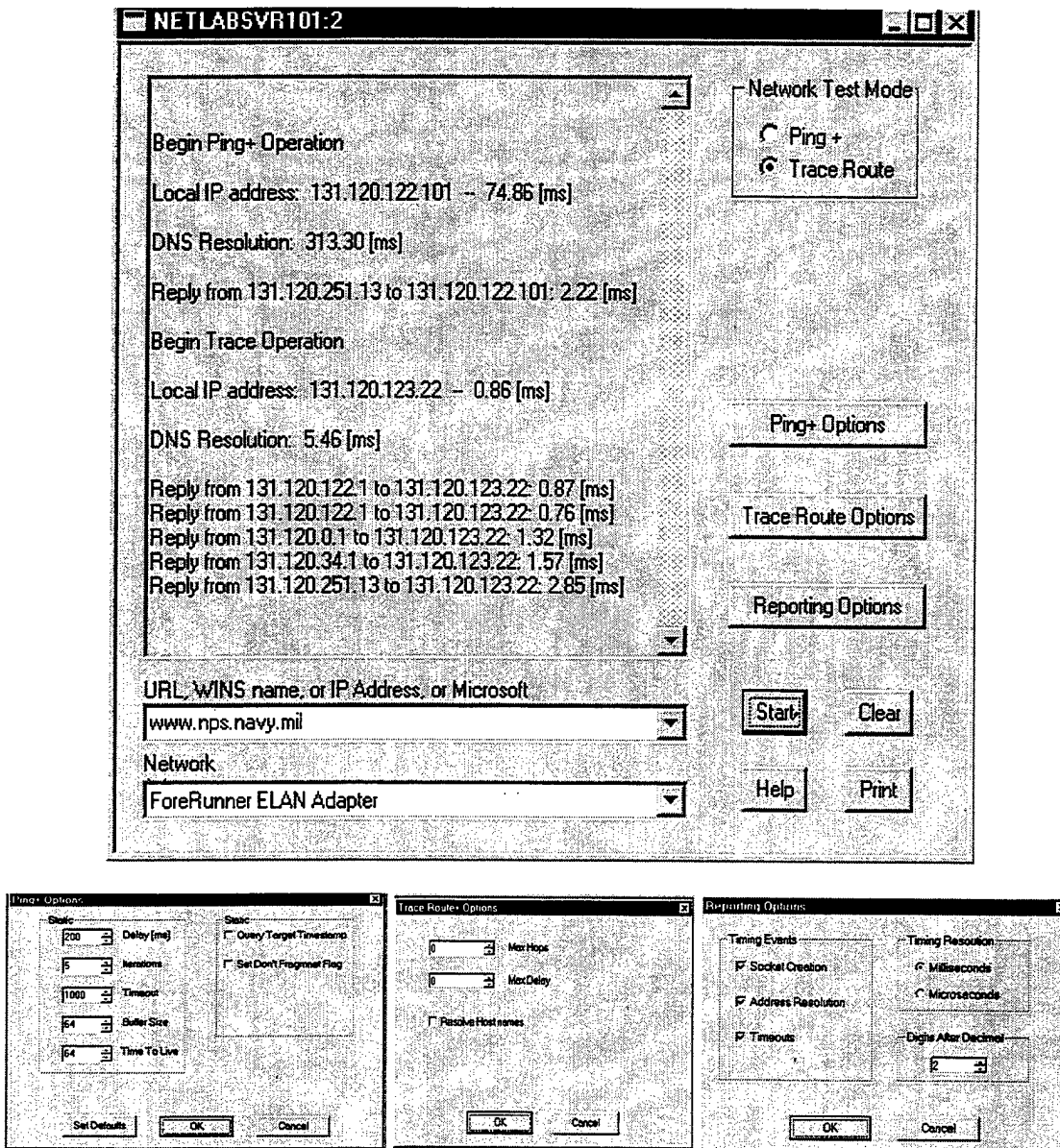


Figure 12: Continuity and Connectivity GUI

The benchmarking GUI is shown in Figure 13. Call set-up times are measured for connection-oriented protocols by timing the operation of the *WSAConnect()* function. This returns the time required for the local socket to establish a connection to a remote socket. Throughput refers to the amount of actual data supplied to the transport layer for transfer. Data rate measurements include the overhead added by the transport and network layers.

These measurements include only the performance of the associated connection; therefore, they provide the best estimate for the networking performance an application will receive. The performance monitor statistics available in Windows™ NT are a measure of network activity and include all network traffic.

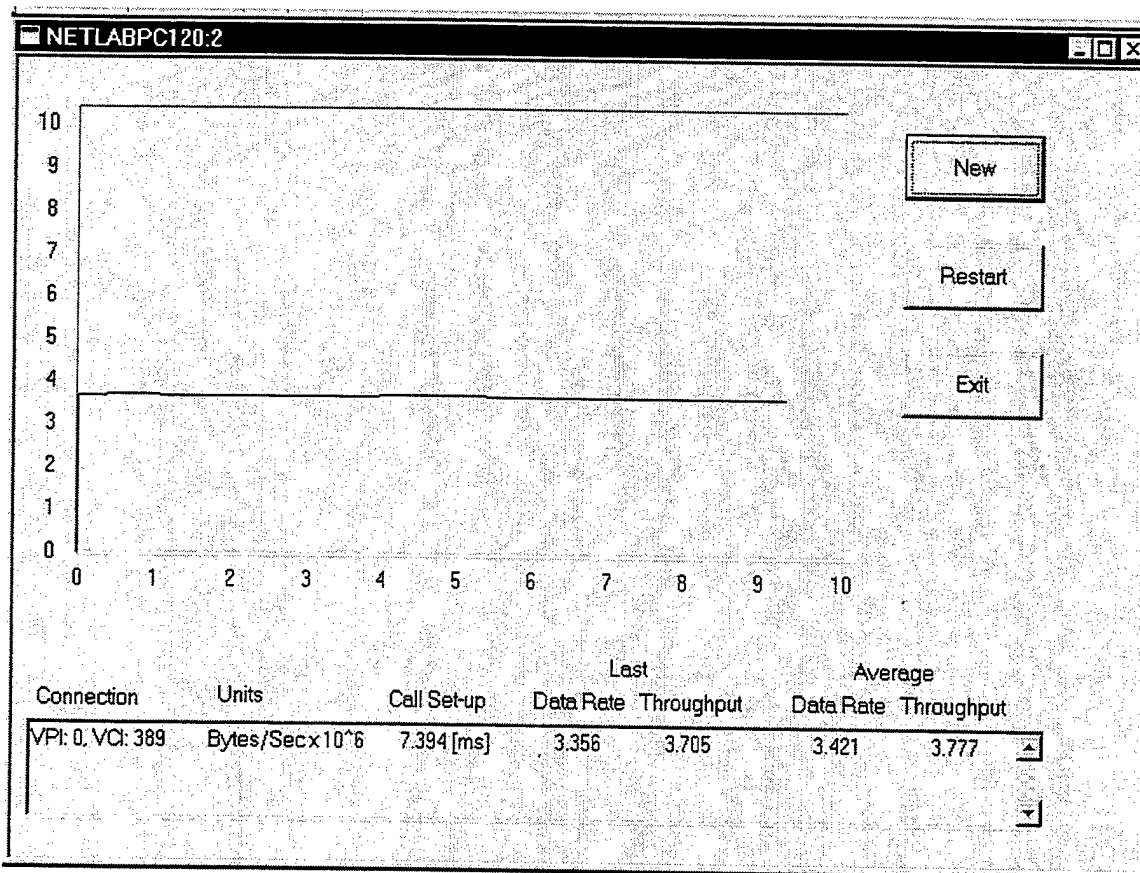
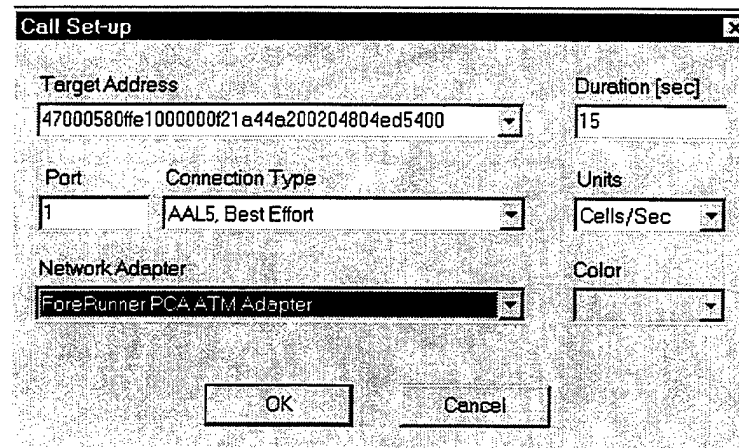


Figure 13: Performance Benchmarking GUI

In order to generalize the performance benchmarking GUI to support different protocols and types of service, the GUI shown in Figure 14 is used to separate the call-setup procedure from the main GUI. This will help in migrating the software to multithreaded operation. In addition, when the software is expanded to support different QoS connections, an additional dialog can be added to allow the parameters specific to that type of connection to be set.



The image shows a 'Call Set-up' dialog box with the following fields and controls:

- Target Address:** A dropdown menu showing the hexadecimal address '47000580fe100000f21a44a200204804ed5400'.
- Duration [sec]:** A text input field containing the value '15'.
- Port:** A text input field containing the value '1'.
- Connection Type:** A dropdown menu showing 'AAL5, Best Effort'.
- Units:** A dropdown menu showing 'Cells/Sec'.
- Network Adapter:** A dropdown menu showing 'ForeRunner PCA ATM Adapter'.
- Color:** A dropdown menu with an empty selection.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom.

Figure 14: Benchmarking Call-Setup GUI

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSION

A. SUMMARY OF WORK

This thesis has thoroughly investigated how WindowsTM NT implements networking support and demonstrated that implementation details are significant to observed network performance. The network analysis software developed is functional and provides a valuable resource for both network administrators and networking researchers. The software provides complete descriptions of a host's networking configuration, significantly improved versions of *ping* and *tracert*, and measures both call-setup time and throughput performance of TCP/IP and ATM AAL5 best effort connections.

B. FUTURE WORK

This network analysis software is structured to facilitate upgrades and expansion. In addition to improving and adding valuable utility, further work will provide more insight into advanced networking implementation and analysis issues. Some specific improvements include:

1. Implementing multi-threaded support. This will allow networking performance to be measured as functions of operating system load and application priority level in addition to raw network throughput.
2. Expanding ATM support to include other AALs.
3. Allowing the user to set QoS parameters for each connection.
4. Providing the ability to remotely view the networking configuration or measure performance benchmarks from any WindowsTM NT machine in that domain.
5. Supporting Telnet, FTP, and rlogin functions.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. INFORMATION TECHNOLOGY FOR THE TWENTY-FIRST CENTURY (IT-21)

RTTUZYUW RUHPGG9842 0890945UUUU—RUWFNAA.

ZNR UUUUU

R 300944Z MAR 97 ZYB PSN 077820Q24

FM CINCPACFLT PEARL HARBOR HI//N00//

TO ALPACFLT

ALLANTFLT

INFO RUENAAA/ASSTSECNAV RDA WASHINGTON DC//C4I//

RUENAAA/CNO WASHINGTON DC//N00/N09/N095/N2/N4/N41/N43/N46/N6/N6B/

N8/N80/N85/N86/N87/N88//

RUCBCLF/CINCLANTFLT NORFOLK VA//N00/N6//

RUCBACM/CINCUSACOM NORFOLK VA//J00/J6//

RHHMUNA/USCINCPAC HONOLULU HI//J00/J6//

RHDLONE/CINCUSNAVEUR LONDON UK//00/N6//

RULSEA/COMNAVSEASYS COM WASHINGTON DC//N00/N08/PMS335/PMS3

RUENMED/BUMED WASHINGTON DC//N00//

RHRMDAB/RUCJNAV/COMUSNAVCENT//N00/N6//

RUCTPOA/CNET PENSACOLA FL//N00//

RUEACNP/BUPERS WASHINGTON DC//N00//

RUHEHMS/COMMARFORPAC//CG/G6//

RUCKMAA/COMMARFORLANT//CG/G6//

RULSSPA/COMSPA WARSYS COM WASHINGTON DC//N00/N05/PMW171/PMW176//

RUWFADO/NAVSTKAIRWARCEN FALLON NV//N00//

RULKSDF/COMNAVSECGRU FT GEORGE G MEADE MD//N00//

RULSAMX/COMNAV SUPSYS COM MECHANICSBURG PA//N00//

RUWFAFK/COMNAV SPECWARCOM CORONADO CA//N00//

RULSADG/NRL WASHINGTON DC//N00//

RULSWCB/COMNAV COMTEL COM WASHINGTON DC//N00/N3//

RUCOSAO/NAVMASSO CHESAPEAKE VA//N00/N6//

RUWFOAA/NCCOSC RDTE DIV SAN DIEGO CA//N433//

RHHMHAH/CINCPACFLT PEARL HARBOR HI//N00//

BT

UNCLAS //N05230//

ALPACFLT 008/97

MSGID/GENADMIN/CINCPACFLT/008//

SUBJ/INFORMATION TECHNOLOGY FOR THE 21ST CENTURY//

POC/M.R. SCOTT/CDR N6/CINCPACFLT//TEL: 808 471-8637//

POC/D.A. STRAUB/CDR N6/CINCLANTFLT//TEL: 757 322-5863//

RMKS/1. THIS IS THE FIRST IN A SERIES OF JOINT CINCPACFLT AND CINCLANTFLT MESSAGES CONCERNING THE DEVELOPMENT AND IMPLEMENTATION OF IT-21. THIS MESSAGE PROVIDES IT-21 HARDWARE/SOFTWARE IMPLEMENTATION STANDARDS FOR PROGRAMS INSTALLING INFORMATION SYSTEMS ON FLEET UNITS/BASES AND PROVIDES THE FLEET WITH GUIDANCE ON MAINTAINING EXISTING INFORMATION SYSTEMS UNTIL INSTALLATION OF IT-21 PRODUCTS. THE IT-21 IMPLEMENTATION STANDARDS OUTLINED BELOW ARE PROMULGATED IN ADVANCE OF DON-WIDE GUIDANCE FROM THE DON CHIEF INFORMATION OFFICER (CIO). THE DON CIO WILL PROMULGATE DON-WIDE STANDARDS FOLLOWING NEGOTIATION OF ENTERPRISE-WIDE NETWORK OPERATING SYSTEMS AND APPLICATIONS.

2. BACKGROUND: INFORMATION SUPERIORITY IS THE FOUNDATION OF JOINT VISION 2010 BATTLEFIELD DOMINANCE, AS WELL AS THE WARFIGHTING VISION FOR EACH SERVICE. NETWORK WARFARE, ROBUST INFRASTRUCTURE AND INFORMATION DISSEMINATION TO DISPERSED FORCES ARE KEY ELEMENTS IN ACHIEVING INFORMATION SUPERIORITY. IT-21 IS A FLEET DRIVEN REPRIORITIZATION OF C4I PROGRAMS OF RECORD TO ACCELERATE THE TRANSITION TO A PC BASED TACTICAL/TACTICAL SUPPORT WARFIGHTING NETWORK. THE INACTIVATION OF THE CURRENT DOD MESSAGING SYSTEM (AUTODIN) BY DEC 99, WITH NO PLANNED NAVY INFRASTRUCTURE REPLACEMENT, MANDATES THE RAPID IMPLEMENTATION OF THIS WARFIGHTING NETWORK.

3. COMMERCIAL NETWORK OPERATING SYSTEMS (NOS) AND E-MAIL PRODUCTS HAVE ACHIEVED FUNCTIONAL PARITY. THE FLEETS CANNOT CONTINUE TO SUPPORT A MULTITUDE OF DIVERSE OPERATING SYSTEMS AND E-MAIL PRODUCTS WITH THEIR OWN TRAINING, OPERATIONAL PROCEDURES AND TROUBLESHOOTING REQUIREMENTS. THE DOD JOINT TECHNICAL ARCHITECTURE (JTA) AND DEFENSE INFORMATION INFRASTRUCTURE COMMON OPERATING ENVIRONMENT (DII COE) PROVIDE DOD WITH THE AIS SYSTEM GUIDANCE REQUIRED TO TAKE THE NAVY INTO THE 21ST CENTURY. THIS CONVERGENCE OF SOLUTIONS, PROBLEMS AND GUIDANCE

PROVIDES THE IMPETUS TO ESTABLISH MINIMUM NAVY AIS STANDARDS AT THIS TIME. IMPLEMENTATION OF THIS POLICY REQUIRES ALL NON-STANDARD NOS AND E-MAIL PRODUCTS BE REPLACED NLT DEC 99.

A. WINDOWS NT SERVER 4.0 IS THE STANDARD FLEET NOS. IT WILL SOON BE FOLLOWED BY WINDOWS NT 5.0. WINDOWS NT SERVER 4.0 IS DII COE COMPLIANT.

B. MS EXCHANGE IS DESIGNATED AS THE STANDARD E-MAIL SOLUTION FOR BOTH FLEETS TO ENSURE AN INTEROPERABLE SECURE MESSAGING SYSTEM IS OPERATIONAL PRIOR TO AUTODIN INACTIVATION NLT DEC 99.

C. MS OFFICE 97 IS DESIGNATED AS THE STANDARD FLEET OFFICE SOFTWARE.

D. EXPENDITURE OF OPERATING FUNDS TO MAINTAIN EXISTING IT-21 NONCOMPLIANT NOS AND APPLICATIONS SHALL BE THE ABSOLUTE MINIMUM NECESSARY TO MEET OPERATING REQUIREMENTS UNTIL IT-21 NOS/SOFTWARE IS INSTALLED EVEN IF TEMPORARY LN DEGRADATION OCCURS. SOFTWARE REQUIREMENTS DRIVE HARDWARE STANDARDS. HARDWARE AND SOFTWARE PURCHASED TODAY MUST BE CAPABLE OF MEETING MISSION REQUIREMENTS THROUGH THE YEAR 2000.

4. CINCPACFLT AND CINCLANTFLT ARE ACTIVELY WORKING WITH OPNAV ON IT-21 FUNDING AND IMPLEMENTATION PLANS. IN GENERAL, AFLOAT IT-21 IMPLEMENTATION WILL BE LINKED TO DEPLOYING BATTLEGROUPS AND ASHORE IT-21 WILL BE IMPLEMENTED IN A PHASED APPROACH. SPECIFIC IMPLEMENTATION SCHEDULES WILL BE PROMULGATED AT A LATER DATE. CINCPACFLT AND CINCLANTFLT ARE TRANSITIONING TO WINDOWS NT 4.0, MS EXCHANGE AND MICROSOFT OFFICE 97. THIS ENVIRONMENT CANNOT BE OPTIMIZED WITHOUT 32 BIT OPERATING SYSTEMS, HIGH RESOLUTION DISPLAYS AND MASS STORAGE. ATM BACKBONE LANS WITH AT LEAST 100 MBS (TCP/IP) TO THE DESKTOP PC WILL BE INSTALLED ON ALL SHIPBOARD LANS, FLEET HEADQUARTERS (CPF, CLF, TYCOMS, GROUP AND SQUADRON COMMANDS) AND SHOULD BE INSTALLED IN THOSE SHORE ACTIVITIES THAT SUPPORT TACTICAL OPERATIONS. THIS WILL THEN ALLOW TRANSITION TO ATM-TO- THE-DESKTOP PC WHEN THE ATM TECHNOLOGY MATURES.

5. SYSTEM COMMANDS AND PROGRAM MANAGERS:

A. NTCSS WILL BECOME THE IT-21 PROGRAM OF RECORD FOR INSTALLATION OF BOTH SECRET AND UNCLASSIFIED LANS ONBOARD COMMISSIONED SHIPS. NTCSS (ATIS/SNAP III) LANS INSTALLED FROM THIS POINT ON WILL HAVE AN ATM BACKBONE, 100 MBS (FAST ETHERNET) TO THE DESKTOP PC AND THE HARDWARE/SOFTWARE OUTLINED AT THE END OF THIS MESSAGE. THE MIGRATION OF NTCSS LANS TO HIGHER CAPACITY LANS WILL REDUCE THE NUMBER OF PC'S DELIVERED DURING INITIAL INSTALLATION. THE TRADE-OFF OF QUANTITY FOR FRONT END PC'S IS REQUIRED TO SUPPORT JV-2010 AND AUTODIN INACTIVATION.

B. SPAWAR IS WORKING WITH NAVSEA TO ENSURE THAT LANS INSTALLED DURING NEW CONSTRUCTION MEET THE IT-21 REQUIREMENTS.

C. APPLICATION PROGRAM MANAGERS SUCH AS JMCIS, NSIPS, TAMPs, AND GCSS SHOULD MIGRATE CURRENT APPLICATIONS TO THE DII COE WITH AN IMMEDIATE OBJECTIVE OF OBTAINING PC WORKSTATION ACCESS TO ALL APPLICATION DATA ON AN ENTERPRISE LAN.

D. PROGRAMS INSTALLING INFORMATION SYSTEMS (NEUNET, SMARTLINK, SMARTBASE, TELEMEDICINE, ETC.) MUST INSTALL COMPONENTS IN FLEET ACTIVITIES THAT MEET IT-21 STANDARDS AND PROVIDE INTEROPERABILITY THROUGHOUT THE WARFIGHTING NETWORK.

6. TYCOMS AND THIRD ECHELON COMMANDS SHALL ENSURE THAT:

A. SHIPS AND ACTIVITIES INSTALLING NEW LANS, UNDERGOING SIGNIFICANT LAN UPGRADES OR THOSE ACTIVITIES WITH STAND ALONE PC'S SHALL INSTALL IT-21 HARDWARE AND SOFTWARE. NEW OR REPLACEMENT SHIPBOARD AND SHORE BASED TACTICAL LANS SHOULD HAVE AN ATM BACKBONE WITH AT LEAST 100 MBS (FAST ETHERNET) TO THE PC.

B. SHIPS AND ACTIVITIES WITH EXISTING LANS, WHICH REQUIRE REPLACEMENT OF UNSERVICEABLE HARDWARE, SHORT OF A FULL NETWORK UPGRADE, SHALL INSTALL HARDWARE WHICH MEETS IT-21 STANDARDS. THE NEW EQUIPMENT MAY NOT BE COMPATIBLE WITH THE EXISTING LAN HARDWARE. CINCPACFLT AND CINCLANTFLT BELIEVE THAT ALL AUTOMATED INFORMATION SYSTEMS (AIS) PROCURED MUST BE COMPATIBLE WITH THE IT-21 LAN STANDARDS EVEN IF TEMPORARY LAN DEGRADATION OCCURS. THERE IS ONLY SUFFICIENT FUNDING TO DO IT RIGHT THE FIRST TIME.

7. THE IT-21 STANDARDS BELOW REPRESENT FRONT END MARKET TECHNOLOGY, ARE DYNAMIC IN NATURE, AND WILL CONTINUE TO BE CLOSELY LINKED TO COMMERCIAL TRENDS. THE STANDARDS LISTED BELOW ARE INTENDED TO BE MINIMUM STANDARDS AND WILL BE UPDATED PERIODICALLY.

A. IT-21 LAN:

(1) AFLOAT LAN STANDARDS - ATM FIBER BACKBONE, 100 MBPS (FAST ETHERNET) TO THE PC.

(2) ASHORE TACTICAL AND HEADQUARTERS COMMAND CENTER STANDARD (CPF, CLF, TYCOMS, GROUP AND SQUADRON COMMANDS) - ATM BACKBONE, 100 MBPS (FAST ETHERNET) TO THE PC.

(3) ASHORE TACTICAL SUPPORT COMMAND STANDARDS (BASES) - ATM BACKBONE, 100 MBPS (FAST ETHERNET) TO THE PC.

(4) METROPOLITAN AREA NETWORKS (MAN) SHOULD BE CAPABLE OF SUPPORTING AT LEAST OC-3 (155MBS).

B. IT-21 SOFTWARE:

- WINDOWS NT 4.0/5.0 WORKSTATION

- MS OFFICE 97 PROFESSIONAL (WORD 97, POWERPOINT 97, EXCEL 97, S ACCESS 97)

- IBM ANTI VIRUS (NAVY LICENSE, AVAIL FROM NAVCIRT)

- MS BACK OFFICE CLIENT

- MS OUTLOOK 97
- MS EXCHANGE 5.0
- MS IMAGE COMPOSER

C. IT-21 DATABASES. RELATIONAL DATABASES THAT CAN SUPPORT WEB TECHNOLOGY LAW THE COE (ORACLE SYBASE, SQL SERVER, ACCESS, ETC.) WILL BE USED TO SUPPORT DATA REQUIREMENTS AND APPLICATION DEVELOPMENT. ALL PROCESS ENGINEERING INITIATIVES THAT RESULT IN DESIGN/REDESIGN OF A DATA COLLECTION/CAPTURE SYSTEM MUST USE COE COMPLIANT RELATIONAL DATABASE MANAGEMENT SYSTEMS (RDBMS) SOFTWARE. THIS REQUIREMENT IS PROVIDED TO ENSURE RDBMS INITIATIVES USE COTS APPLICATION SOFTWARE. FOR ADDITIONAL INFORMATION ON RELATIONAL DATABASES CONTACT CDR SANDY BUCKLES, CPF N67, COMM/DSN (808) 474-6384, NIPRNET <mailto:U67@CPF-EMH.CPF.NAVY.MIL>.

D. MINIMUM IT-21 PC CAPABILITIES: CPF CAN CURRENTLY PURCHASE THE IT-21 STANDARD PC WITH SOFTWARE FOR \$3250.00 - \$3579.00 - SEE PARA 7(H) AND 7(I).

- 200 MHZ PENTIUM PRO CPU
- 64 MB EDO RAM
- 3.0 GB HARD DRIVE
- 3.5 INCH FLOPPY DISK DRIVE
- 8X IDE CD-ROM
- DUAL PCMCIA/PC CARD READER
- PCI VIDEO W/2MB RAM
- 17 INCH MONITOR (1280 X 1024)
- POINTING DEVICE (TRACKBALL OR MOUSE)
- SOUNDBLASTER (COMPATIBLE) AUDIO CARD WITH SPEAKERS KEYBOARD
- CPU COMPATIBLE 100 MBPS FAST ETHERNET NIC

E. STANDARD IT-21 LAPTOP WORKSTATION: APPROXIMATELY \$5300-
SEE PARA 7(H).

- 150 MHZ PENTIUM
- 32 MB EDO RAM
- 12.1 IN SVGA ACTIVE MATRIX COLOR DISPLAY
- 2.1 GB EIDE HDD
- 6X INTERNAL CD-ROM
- MODEM, PCMCIA SLOTS, NIC CARD
- SMART LITHIUM BATTERY

F. IT-21 NT FILE SERVER FOR DIRECTORY NETWORK SERVICE: APPROXIMATELY \$26K- SEE PARA 7(H). THESE ARE MINIMUM SPECIFICATIONS. NEEDS OF THE SPECIFIC NETWORK WILL DICTATE REQUIREMENTS.

- DUAL 166 MHZ PENTIUM CPU
- 512K SECONDARY CACHE MEMORY- 256 MB RAM
- TWO 4 GB SCSI HDD
- ONE 6 GB DAT DRIVE
- ONE 3.5 INCH FLOPPY DISK DRIVE
- 6X SCSI CD-ROM
- DUAL PCMCIA/PC CARD READER
- 2 DPT SCSI III CACHING CONTROLLERS (SMARTCACHE 4)
- PCI VIDEO W/2MB RAM
- 17 INCH MONITOR (1280 X 1024)
- POINTING DEVICE (TRACKBALL OR MOUSE)
- KEYBOARD
- TWO CABLETRON CPU COMPATIBLE ATM NIC CARDS
- ANTEC DUAL POWER SUPPLY CASE (HOT SWAPPABLE)

G. IT-21 FILE SERVER/APPLICATION SERVER: APPROXIMATELY \$26K- SEE PARA 7(H). SAME AS IT-21 NT FILE SERVER FOR DIRECTORY NETWORK SERVICE WITH THE FOLLOWING CHANGES:

- CHANGE HDD RQMT TO FIVE 4 GB DRIVES
- CHANGE DAT TO 18 GB.

H. PRICES FOR PC TECHNOLOGY ARE CONSTANTLY CHANGING AND CAN VARY GREATLY DEPENDING ON METHOD OF PROCUREMENT. FOR EXAMPLE, ON 28 MAR 97 AN IT-21 PC PURCHASED DIRECTLY FROM A VENDOR COSTS \$3643. GOVERNMENT RATE FOR SMALL PURCHASES (LESS THAN TEN) IS \$3579. A BULK PROCUREMENT (MORE THAN SEVENTY-FIVE) COSTS \$3250. THE ABOVE PRICES INCLUDE SHIPPING. BULK PROCUREMENTS SHOULD BE MADE THROUGH THE TYPE COMMANDERS WHEN APPROPRIATE. MR. RICK KOOKER, CPF N65, COMM/DSN(808) 474-5882, NIPRNET: U65@CPF-EMH.CPF.NAVY.MIL IS AVAILABLE TO ASSIST TYCOMS WITH AIS PROCUREMENT ISSUES.

I. AS NETWORK COMPUTER TECHNOLOGY EVOLVES SOME COMMANDS MAY BE ABLE TO TRANSITION TO NETWORK COMPUTERS. WHEN CONSIDERING INSTALLATION OF NETWORK COMPUTERS, TOTAL NETWORK COST MUST BE EVALUATED. NETWORK COMPUTERS HAVE NOT MATURED SUFFICIENTLY TO IMPLEMENT THEM IN FLEET PLATFORMS AT THIS TIME.

8. WAIVER REQUESTS FROM THE ABOVE STANDARDS SHOULD BE SUBMITTED DIRECTLY TO THE RESPECTIVE CPF/CLF N6. POINTS OF CONTACT ARE AS FOLLOWS:

- A. CINCLANTFLT: CDR DEBRA STRAUB AT COMM (757) 3225863, NIPRNET: <mailto:U6@CLF.NAVY.MIL>
- B. CINCPACFLT: CDR MIKE SCOTT AT COMM (808) 4747860, NIPRNET: U6@CPF-EMH.CPF.NAVY.MIL //

BT
#9842
NNNN

APPENDIX B. HUNGARIAN NOTATION

Hungarian notation uses prefixes to identify a variable's type and is widely used in Windows coding. There are many, often incompatible, dialects. Following are the conventions I found to be most common and used in developing my code.

<u>Prefix</u>	<u>Corresponding type</u>
a	array
b	Boolean
c	count
C	class
ch	char (NOT equal to a byte if UNICODE or MBCS is defined)
ctl	Windows control
d	double
dw	DWORD
f	float (Another common usage is for flag i.e. a Boolean variable)
g_	global
h	handle
i	int
i64	LARGE_INTEGER
l	long
lp	far *
lst	list box
m_	member variable
ms_	member static
mnu	menu
n	int (Microsoft™ code will use this for uint variables too)
o	object
p	pointer
s_	static
str	CString
sz	NULL terminated string
txt	text control
u	unsigned
v	void
w	word
WM_	Windows message
wnd	window

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. CLASS OVERVIEW

CAboutDialog

Derived from *CDialog*, this class implements the *Help*→*About* menu selection.

CAdapterFormView

Derived from *CFormView*, this class displays network adapter parameters in a dialog resource.

CAdapterSplitWnd

Derived from *CMDIChildWnd*, this class manages the window used to display the network adapter parameters. It includes a *CSplitterWnd*, *CAdapterTreeView*, and *CAdapterFormView* objects.

CAdapterTreeView

Derived from *CTreeView*, this class manages the tree control and organizes installed adapter drivers by NIC.

CGraphCtl

Derived from *CStatic*, this class plots line graphs in the area defined by a *static* dialog control. It relies on *CObservedPerformanceArray* for scaling.

CIcmpPacket

This class encapsulates the tasks for working with ICMP messages.

CMainFrame

Derived from *CMDIFrameWnd*, this class manages the menu bar and all child windows.

CNetBenchFormView

Derived from *CFormView*, this class implements the performance benchmarking GUI. It requires *CPerformanceSocket*, *CWeightedInterval*, and *CGraphCtl*.

CNetManApp

Derived from *CApplication*, this class manages the application primary thread.

CNetManDoc

Derived from *CDocument*, this class retrieves configuration data from the registry and provides that data to any *CView* derived class that requests it.

CObservedPerformanceArray

This class stores an array of *CWeightedInterval* objects and translates the results to a given resolution. This class provides data that can be directly used by *CGraphCtl*.

CPerformanceSocket

Derived from *CAsyncSocket*, this class provides full socket functionality and returns a *CWeightedInterval* describing the execution time.

CPingFormView

Derived from *CFormView*, this class implements the continuity and connectivity (*ping* and *tracert*) GUI. It uses *CPerformanceSocket*, *CTraceOptionsDialog*, *CPingOptionsDialog*, and *CReportingOptionsDialog*.

CPingOptionsDialog

Derived from *CDialog*, this class encapsulates the dialog used to set the user-selectable options for the *ping* mode of operation.

CProtocolFormView

Derived from *CFormView*, this class displays the the information about a selected protocol. It is has been replaced by *CProtocolPropertySheetView*.

CProtocolPropertyPageGeneral

Derived from *CPropertyPage*, this class handles the dialog associated with the *General* tab of the protocol property sheet.

CProtocolPropertyPageServiceFlags

Derived from *CPropertyPage*, this class handles the dialog associated with the *Service Flags* tab of the protocol property sheet.

CProtocolPropertyPageSocketParameters

Derived from *CPropertyPage*, this class handles the dialog associated with the *Socket Parameters* tab of the protocol property sheet.

CProtocolPropertyPageUtil

Derived from *CPropertyPage*, this class handles the dialog associated with the *Utilities* tab of the protocol property sheet. The access to these utilities will be moved to the main menu bar in future versions.

CProtocolPropertySheet

Derived from *CPropertySheet*, this class handles the protocol property sheet control. It contains instances of the following classes: *CProtocolPropertyPageGeneral*, *CProtocolPropertyPageServiceFlags*, and *CProtocolPropertyPageSocketParameters*.

CProtocolProertySheetView

Derived from *CView*, this class provides *CView* functionality to *CProtocolPropertySheet*. Multiple inheritance is not used because of MFC limitations.

CProtocolSplitWnd

Derived from *CMDIChild*, this class contains the *CSplitter* object. It also includes a global variable to facilitate communication between *CProtocolTreeView* and *CProtocolPropertySheet*.

CProtocolTreeView

Derived from *CTreeView*, this class manages the tree control and organizes installed protocols by service provider.

CReportingOptionsDialog

Derived from *CDialog*, this class encapsulates the dialog used to set the user-selectable timing options.

CTraceOptionsDialog

Derived from *CDialog*, this class encapsulates the dialog used to set the user-selectable options for the *trace* mode of operation.

CWeightedInterval

This class contains a start time, stop time, and a weight value. This application uses the weight to represent a number of bytes. This class will return data in a number of different formats, including *CPoint*(midpoint, weight); *CPoint*(endpoint, weight); and *CPoint*(endpoint, weight/range).

LIST OF REFERENCES

- [1] MSG, "*Information Technology for the 21st Century*," DTG 300944ZMAR97, CINCPACFLT Pearl Harbor, HI. 1997.
- [2] Raj Rajagopal and Subodh P. Monica, "*Windows NT4 Advanced Programming*," Berkeley, CA: Osborne/McGraw-Hill, 1998.
- [3] Stevens, W. Richard, "*TCP/IP Illustrated, Volume 1: The Protocols*," Reading, MA: Addison-Wesley, 1988.
- [4] Stevens, W. Richard, "*TCP/IP Illustrated, Volume 2: The Implementation*," Reading, MA: Addison-Wesley, 1988.
- [5] Schwartz, Mischa, "*Broadband Integrated networks*," Upper Saddle River, NJ: Prentice Hall, 1997.
- [6] "Microsoft Foundation Class Library Guidelines," MSDN, 1997.
- [7] "REG: Network Adapter Cards Entries, PART 1," Article ID: Q102999, MSDN, 1997.
- [8] "REG: Network Adapter Cards Entries, PART 2," Article ID: Q102994, MSDN, 1997.
- [9] Chu, Johnson, Moore, Treadwe, "Write an NT WinSock Service," <http://www.byte.com>, December, 1994.
- [10] W. Stallings, "*Data and Computer Communications*," Upper Saddle River, NJ: Prentice Hall, 1997.
- [11] <http://www.stardust.com>, Penton Media, Inc., Campbell, California.
- [12] Fore Systems, "*ForeRunner HE/200 E/LE ATM Adapters for the PC User Manual*," 1995-1999, Fore Systems, Warrendale, PA, 1995.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center 8725 John J. Kingman Rd., STE 0944 Ft. Belvoir, VA 22060-6218	2
2. Dudley Knox Library..... Naval Postgraduate School 411 Dyer Rd. Monterey, CA 93943-5101	2
3. Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
4. Curricular Officer, Code 34 Engineering and Technology Naval Postgraduate School Monterey, CA 93943-5109	1
5. Professor John McEachen, Code EC/Mj Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
6. Professor Murali Tummala, Code EC/Tu Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5121	1
7. Ms. Rosemary Wenchel..... Naval Information Warfare Analysis Center 9800 Savage Road Ft. Meade, MD 20755	1
8. Commanding Officer Naval Information Warfare Activity 9800 Savage Road Ft. Meade, MD 20755	1

9. Lieutenant David R. DeMille 2
Naval Station Ingleside
125 Coral Sea Rd
Trailer 13
Ingleside, TX 78362